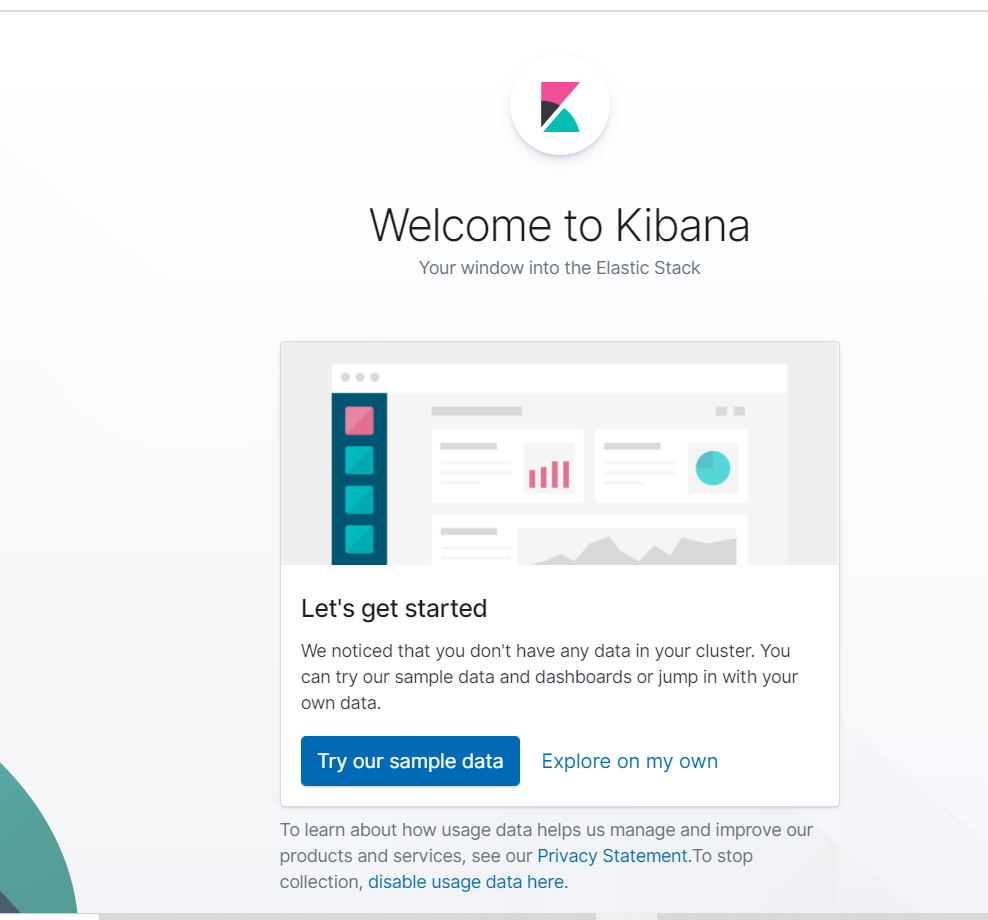
**## Activity File: Exploring Kibana**

​

\* You are a DevOps professional and have set up monitoring for one of your web servers. You are collecting all sorts of web log data and it is your job to review the data regularly to make sure everything is running smoothly.

​

\* Today, you notice something strange in the logs and you want to take a closer look.

​

\* Your task: Explore the web server logs to see if there's anything unusual. Specifically, you will:

​

:warning: **\*\*Heads Up\*\***: These sample logs are specific to the time you view them. As such, your answers will be different from the answers provided in the solution file.

​

---

​

**### Instructions**

​

1. Add the sample web log data to Kibana.

​

2. Answer the following questions:

​

- In the last 7 days, how many unique visitors were located in India?

236

​

- In the last 24 hours, of the visitors from China, how many were using Mac OSX?

7

​

- In the last 2 days, what percentage of visitors received 404 errors? How about 503 errors?

404: 100%

503: 0%

- In the last 7 days, what country produced the majority of the traffic on the website?

CN

- Of the traffic that's coming from that country, what time of day had the highest amount of activity?

10 am and 12 pm

- List all the types of downloaded files that have been identified for the last 7 days, along with a short

description of each file type (use Google if you aren't sure about a particular file type).

Gz files: A GZ file is an archive file compressed by the standard GNU zip (gzip)

Css files: A CSS file is a cascading style sheet ([CSS](https://techterms.com/definition/css)) file used to format the contents of a webpage

Deb files: deb file is used to signify a collection of files managed by the Debian packages management system

Zip files: ZIP is an archive file format that supports lossless data compression

Rpm files: RPM originally stands for Red Hat Package Manager, is a free and open-source package management system

​

3. Now that you have a feel for the data, Let's dive a bit deeper. Look at the chart that shows Unique Visitors Vs. Average Bytes.

- Locate the time frame in the last 7 days with the most amount of bytes (activity).

3 hours

- In your own words, is there anything that seems potentially strange about this activity?

It is suspicious that the average amount of bytes transferred doubled from the previous activity from 8,682 to 15,709

​

4. Filter the data by this event.

- What is the timestamp for this event?

|  |  |
| --- | --- |
|  |  |

2021-05-23 21:00 May 23, 2021 @ 22:57:28.552

- What kind of file was downloaded?

rpm

- From what country did this activity originate?

India

- What HTTP response codes were encountered by this visitor?

200

​

5. Switch to the Kibana Discover page to see more details about this activity.

- What is the source IP address of this activity?

35.143.166.159

- What are the geo coordinates of this activity?

geo.coordinates:

{ "lat": 43.34121, "lon": -73.6103075 }

- What OS was the source machine running?

Win 8

- What is the full URL that was accessed?

https://artifacts.elastic.co/downloads/beats/metricbeat/metricbeat-6.3.2-i686.rpm

- From what website did the visitor's traffic originate?

​ http://facebook.com/success/jay-c-buckey

6. Finish your investigation with a short overview of your insights.

​

- What do you think the user was doing?

The user is trying to masquerade himself and make the victim appears to be the imposter.

- Was the file they downloaded malicious? If not, what is the file used for?

No, the file is not malicious. The downloaded file I686.rpm is an rpm file. The name RPM variously refers to the .rpm file format, files in this format, software packaged in such files, and the package manager itself.

- Is there anything that seems suspicious about this activity?

Yes, “request and GET and 200” tells us that something successfully happened

- Is any of the traffic you inspected potentially outside of compliance guidelines?

**## Activity File: Kibana Continued**

​

- This week, you created the infrastructure behind a security information and event management system such as Kibana. Once that set up is complete, you will have finished the project.

​

- This optional activity tasks you with exploring more Kibana capabilities, some of which you will use in future projects.

​

- **\*\*Note\*\***: In order to complete these activities, you will need to complete the optional Metricbeat configuration.

​

​

**### Scenario**

​

In this activity, you will suppose the role of a cloud architect that has been tasked with setting up an ELK server to gather logs for the Incident Response team.

​

Before you hand over the server to the IR team, your senior architect has asked that you verify the ELK server is working as expected and pulling both logs and metrics from the pen-testing web servers.

​

You will have three tasks:

​

1. Generate a high amount of failed SSH login attempts and verify that Kibana is picking up this activity.

​

2. Generate a high amount of CPU usage on the pen-testing machines and verify that Kibana picks up this data.

​

3. Generate a high amount of web requests to your pen-testing servers and make sure that Kibana is picking them up.

​

​

These activities will guide you though generating some data to visualize in Kibana. Each of these activity will require the following high level steps:

​

1. Use your jump-box to attack your web machines in various ways.

2. Use a Linux utility to stress the system of a webVM directly.

3. Subsequently generate traffic and logs that Kibana will collect.

4. View that traffic in various ways inside Kibanna.

​

​

It's also worth noting that these activities comprise different job roles:

​

- Getting the infrastructure setup and maintaining it is the role of a security engineer or cloud architect.

​

- Using that infrastructure by creating dashboards and alerts fall under the security analyst role. It would be rare to have a position where you would be required to do both.

​

That said, now that we have Kibana setup and gathering data from three web servers, its worth learning how to visualize data in Kibana.

​

Before getting started, we'll have to complete some metrics and logs set up.

​

**### Setup: Kibana Metrics and Logs Orientation**

​

Before we begin generating traffic, locate the two screens inside Kibana that you will use to visualize this traffic:

​

- Logs

- Metrics

​

![](Images/Metrics-Logs.png)

​

These pages will show you the changes in data that we will create.

​

**#### Logs**

​

- Click **\*\*Logs\*\*** to see some general system logs coming from the web machines.

​

![](Images/Logs-General.png)

​

- Notice that you can stream logs live from the machines.

​

![](Images/Stream-Live.png)

​

**#### Metrics**

​

- Next, click **\*\*Metrics\*\*** on the left side.

​

- Here we can see each of our VMs that are sending metrics.

​

- Click on one of the large squares that represent one of your VMs.

​

- Choose **\*\*View metrics\*\*** from the dropdown that appears.

​

![](Images/Metric-VM-Dropdown.png)

​

- Notice that you can see CPU and memory usage here.

​

![](Images/Host-Overview.png)

​

Now that we know where to look for this data, let's generate some unusual network traffic.

​

**### Activity Tasks**

​

Expand the provided activity files to complete each task. These tasks can be completed in any order.

​

**#### SSH Barrage**

​

Task: Generate a high amount of failed SSH login attempts and verify that Kibana is picking up this activity.

​

<details>

<summary> Activity File: SSH Barrage </summary>

​

#### Scenario

​

- You are a cloud architect that has been tasked with setting up an ELK server to gather logs for the Incident Response team to use for training.

​

- Before you hand over the server to the IR team, your senior architect has asked you to verify the ELK server is working as expected and pulling both logs and metrics from the pentesting web servers.

​

\*\*Your Task\*\*: Generate a high amount of failed SSH login attempts and verify that Kibana is picking up this activity.

​

---

​

#### Instructions

​

One way we can generate logs of interest is to create some failed SSH logins on our servers.

​

- The only environment that holds our SSH keys is our Ansible container. Attempting to create an SSH connection from any other environment will trigger a log entry.

​

- We can also create a log entry by attempting to log in with the wrong username.

​

- Note: A successful SSH login also creates a log entry, but here we will focus on failed logins.

​

We can easily do this by trying to SSH to a web machine from our jump box directly without using the Ansible container.

​

1. Start by logging into your jump-box.

​

- Run: `ssh username@ip.of.web.vm`

​

- You should receive an error:

​

```bash

sysadmin@Jump-Box-Provisioner:~$ ssh sysadmin@10.0.0.5

sysadmin@10.0.0.5: Permission denied (publickey).

```

​

- This error was also logged and sent to Kibana.

​

2. Run the failed SSH command in a loop to generate failed login log entries.

​

- You can use a bash `for` or `while` loop, directly on the command line, to repeatedly run the SSH command.

for i in {1..10}; do ssh [azureuser@10.0.0.5](about:blank); done

​

3. Search through the logs in Kibana to locate your generated failed login attempts.

​

\*\*Bonus\*\*: Create a nested loop that generates SSH login attempts across all three of your VM's.

​ for i in {1..10}; do ssh azureuser@10.0.0.5; ssh azureuser@10.0.0.6; ssh azureuser@10.0.0.7; done

​

​

</details>

​

**#### Linux Stress**

​

Task: Generate a high amount of CPU usage on the pentesting machines and verify that Kibana picks up this data.

​

<details>

​

<summary> Activity File: Linux Stress </summary>

​

​

#### Scenario

​

- You are a cloud architect that has been tasked with setting up an ELK server to gather logs for the Incident Response team to use for training.

​

- Before you hand over the server to the IR team, your senior architect has asked that you verify the ELK server is working as expected and pulling both logs and metrics from the pen-testing web servers.

​

​

\*\*Your Task\*\*: Generate a high amount of CPU usage on the pentesting machines and verify that Kibana picks up this data.

​

---

​

#### Notes

​

The Metrics page for a single VM shows the CPU usage for that machine. This shows how much work the machine is doing. Excessively high CPU usage is typically a cause for concern, as overworked computers are at greater risk for failure.

​

- Metricbeat forwards data about CPU load to Elasticsearch, which can be visualized with Kibana.

​

- In this activity, you will intentionally stress the CPU of one of your VMs, then find evidence of the increased activity in Kibana.

​

Linux has a common, easy-to-use diagnostic program called `stress`. It is easy to use and can be downloaded via `apt`.

​

#### Instructions

​

1. From your jump box, start up your Ansible container and attach to it.

​

2. SSH from your Ansible container to one of your WebVM's.

​

3. Run `sudo apt install stress` to install the stress program.

​

4. Run `sudo stress --cpu 1` and allow `stress` to run for a few minutes.

​

5. View the Metrics page for that VM in Kibana. What indicates that CPU usage increased?

​

6. Run the `stress` program on all three of your VMs and take screenshots of the data generated on the Metrics page of Kibana.

​

- \*\*Note:\*\* The stress program will run until you quit with Ctrl+C.

</details>

​

​

**#### wget-DoS**

​

​

Task: Generate a high amount of web requests to your pen-testing servers and make sure that Kibana is picking them up.

​

<details>

​

<summary> Activity File: wget-DoS </summary>

​

​

#### Scenario

​

- You are a cloud architect that has been tasked with setting up an ELK server to gather logs for the Incident Response team to use for training.

​

- Before you hand over the server to the IR team, your senior architect has asked that you verify the ELK server is working as expected and pulling both logs and metrics from the pen-testing web servers.

​

\*\*Your Task\*\*: Generate a high amount of web requests to your pen-testing servers and make sure that Kibana is picking them up.

​

---

​

#### Instructions

​

The Metrics section for a single VM will show Load and Network Traffic data.

​

We can generate abnormal data to view by creating a DoS web attack. The command-line program `wget` can do this easily.

​

`wget` will download a file from any web server. Use man pages for more info on `wget`.

​

1. Log into your jump box.

​

2. Run `wget ip.of.web.vm`.

​

```bash

sysadmin@Jump-Box-Provisioner:~$ wget 10.0.0.5

--2020-05-08 15:44:00-- http://10.0.0.5/

Connecting to 10.0.0.5:80... connected.

HTTP request sent, awaiting response... 302 Found

Location: login.php [following]

--2020-05-08 15:44:00-- http://10.0.0.5/login.php

Reusing existing connection to 10.0.0.5:80.

HTTP request sent, awaiting response... 200 OK

Length: 1523 (1.5K) [text/html]

Saving to: ‘index.html’

​

index.html 100%[=======================>] 1.49K --.-KB/s in 0s

​

2020-05-08 15:44:00 (179 MB/s) - ‘index.html’ saved [1523/1523]

```

​ # wget 10.0.0.5

3. Run `ls` to view the file you downloaded from your web VM to your jump box.

​

```bash

sysadmin@Jump-Box-Provisioner:~$ ls

index.html

```

​

4. Run the `wget` command in a loop to generate many web requests.

​

- You can use a bash `for` or `while` loop, directly on the command line, just as you did with the SSH command.

​

for i in {1..20}; do wget 10.0.0.5; done

5. Open the Metrics page for the web machine you attacked and answer the following questions:

- Which of the VM metrics were affected the most from this traffic?

​Network Traffic

Memory usage

CPU usage

\*\*Bonus\*\*: Notice that your `wget` loop creates a lot of duplicate files on your jump box.

​

- Write a command to delete \_all\_ of these files at once.

​ find . -name "\*.html" -type f or rm index.\*

- Find a way to run the `wget` command without generating these extra files.

for i in {1..20}; do wget -p 10.0.0.5; done

- Look up the flag options for `wget` and find the flag that lets you choose a location to save the file it downloads.

wget –P [destination\_directory] [URL] wget -P wgetdir 10.0.0.5

- Save that file to the Linux directory known as the "void" or the directory that doesn't save anything.

​ wget -P ~/dev/null 10.0.0.5

\*\*Bonus\*\*: Write a nested loop that sends your `wget` command to all three of your web VMs over and over.

wget 10.0.0.5 10.0.0.6 10.0.0.7

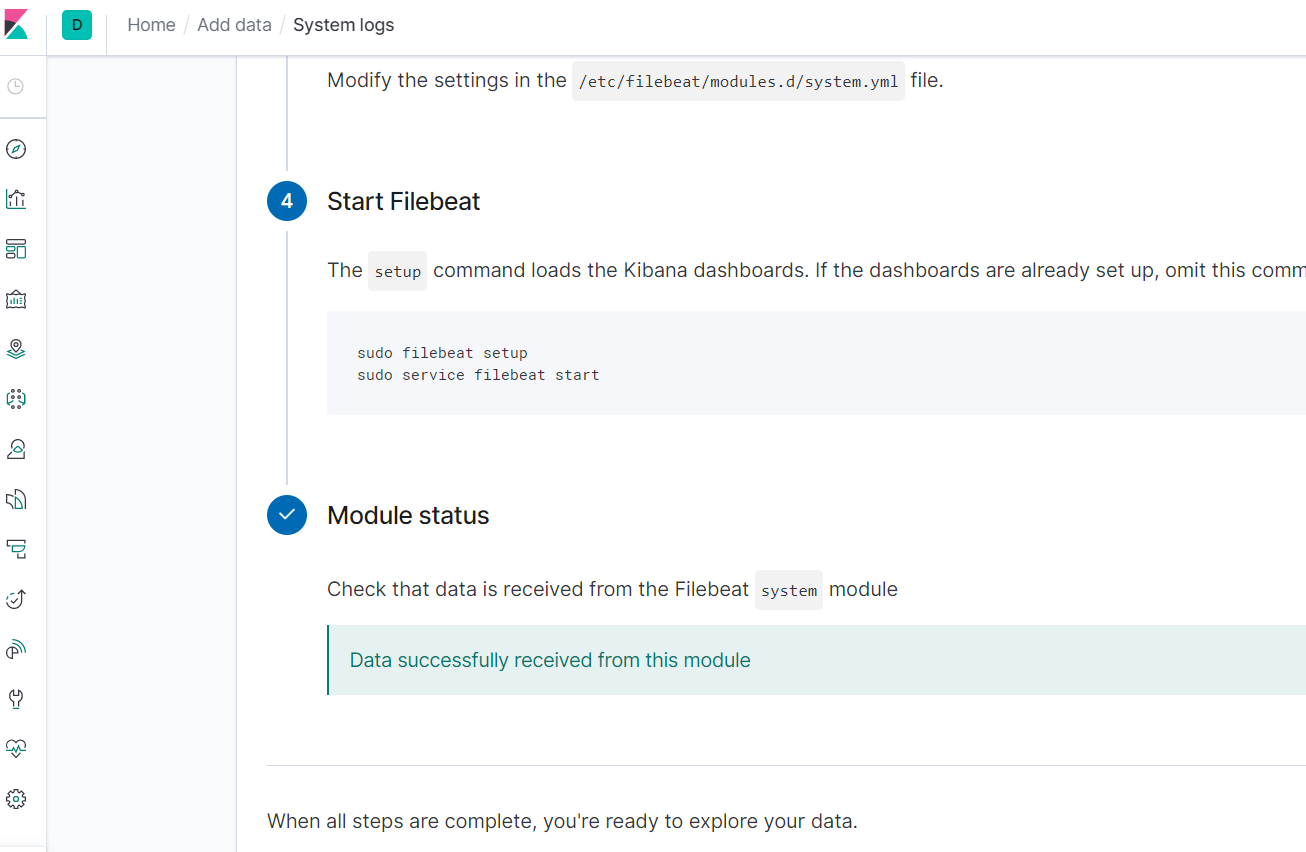
​

</details>

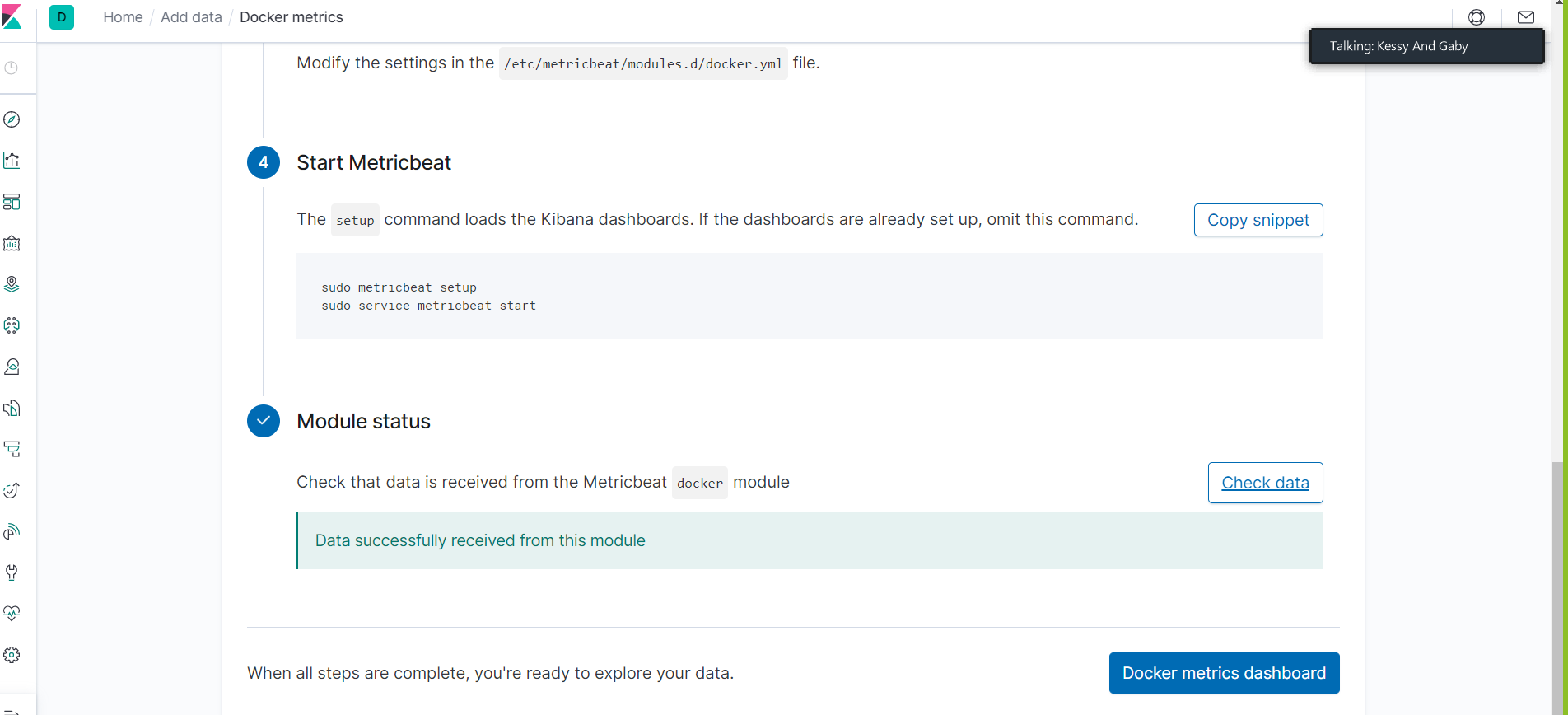
\*\*\*\*Above, you can find the image of :

Filebeat: which is an agent on the servers. It monitors the log files. Collects log events and forward them to Elastic search or logstash.

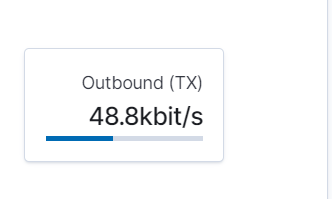
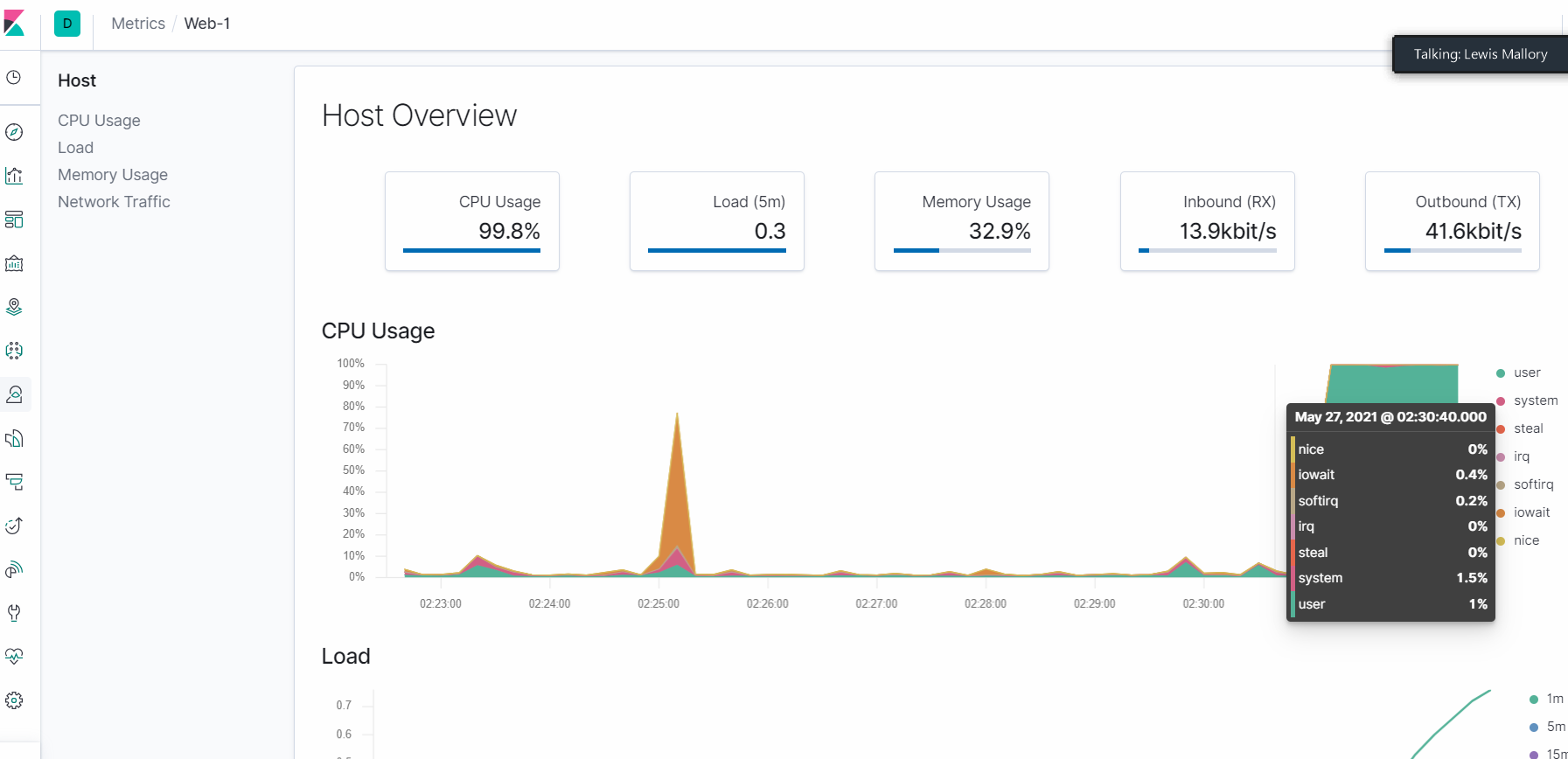
We used filebeat with Elk to be able to transfer data.



You can also find the Metricbeat: which is a tool that can be installed on a target server to collect metrics from O.S. It’s also used to monitor other beats and Elk Stack itself.



We were able to find out that the most traffic occurred here:



Here is the stress web:

